Vision-based detection of non-cooperative UAVs using frame differencing and temporal filter

Christoph Briese, Andreas Seel
Unmanned Aircraft, Institute of Flight Systems
German Aerospace Center (DLR),
Braunschweig, Germany
christoph.briese@dlr.de

Franz Andert
Institute of Transportation Systems
German Aerospace Center (DLR),
Berlin, Germany

***Abstract*— In this paper, we introduce a fast and lightweight method based on several combined filters to detect and track an object in images recorded by a moving camera. Assuming we know nothing about the intruders shape, color or other geometric appearance, we focus with our work on change detection in the image, caused by movement of the object against the background. The method is evaluated with image data from experimental flights with two unmanned aircraft performing different flight maneuvers. The correctness of the intruder detection is evaluated by comparison with hand labeled ground truth from different sequences of the test flight. Additionally, we evaluate the performance of our implementation on architectures with low computational power with regard to a practical onboard solution for small unmanned aerial vehicels (UAV).**

I. Introduction

The significant increase of ready-to-fly drones affects safety of airspace and critical ground infrastructure as well as security and privacy issues in areas wherever third-party interests are to be considered. Geo-fencing solutions and upcoming regulations are going to cope with many accidental problems, however, there are threats arising from intentional attacks, e.g. remotely piloted espionage and terrorism. With that, drone defense becomes an important field in research and development.

There are already several methods to cope with small flying intruders. One is to detect and localize the aircraft or the remote pilot with stationary or vehicle-based protection systems. Once detected, navigation or control signals are jammed or spoofed so that a drone will at least not be able to fly into the protected area. Practical solutions are also handheld devices e.g. presented in [4]. Another category of non-dangerous and thus generally applicable solutions is mainly to catch them e.g. with net throwers [2]. However, if the protected area is not easily or fast accessible from ground or by other ad-hoc solutions, active protection becomes difficult. Spoofing or jamming radio signals can also be critical because it is restricted by the law or would tie down own protectable infrastructure. Thus, the idea is to develop a small UAV capable of intercepting and hunting intruders. Before it comes to a final solution and the discussion of a defense strategy - such a system is ideally autonomous and able to approximate to any intruder. The proposed solution is a patrolling UAV scanning the environment, and once an intruder has been detected, the UAV will stick and approximate.

Fig. 1: Image from the defenders on-board camera while the intruder is hovering in front.

II. Background and Related Work

We consider the following scenario where an automated camera drone (called: *defender*) surveying for moving intruders. The focus of this paper is the detection and tracking of small moving objects (called: *intruders*) within the cameras field of view. From the sensing perspective, the problem is highly related to aircraft collision avoidance and also ground object tracking, although some specific differences will arise. In contrast to most ground object tracking solutions, moving objects of interest are rather small (i.e. an approaching intruder has to be detected as early as possible), and in contrast to typical sense-and-avoid situations with larger aircraft, optical background motion is generally higher (i.e. the defender can move and turn fast if required). Additionally, image background is highly variable, since the intruder can be visually above or below horizon. At this stage of research, image processing is assumed to be comparable to visual

Thanks to the Unmanned Aircraft Team at DLR, namely Michael Kislat-Schmidt, Gordon Strickert, Jan Binger and Endre-Aladar Papp for performing and assisting the helicopter flight operation and Stefan Krause for assist on log data conversion

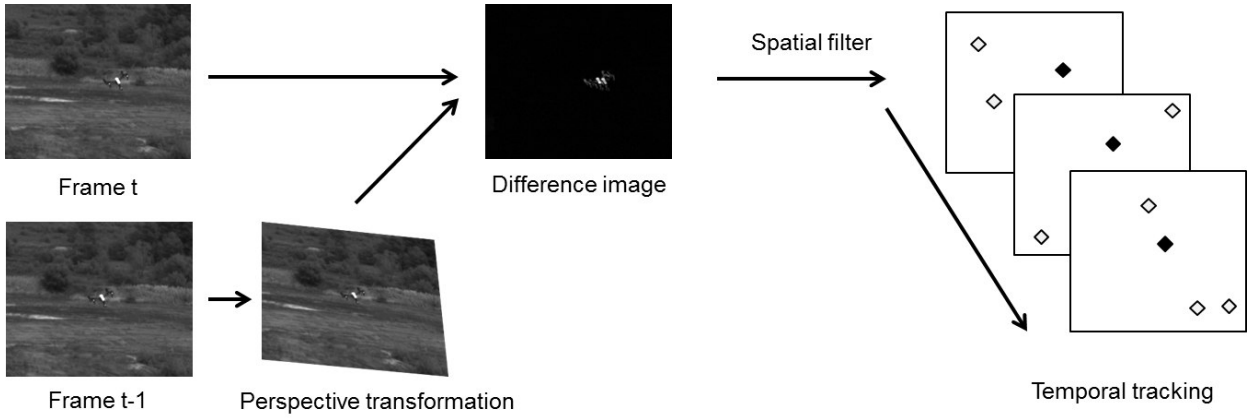


Fig. 2: Workflow of the proposed algorithm.

sense-and-avoid, however, some specific situations will be analyzed.

Within related sense-and-avoid research, radar is one of the major sensors for long-distance detection at larger vehicles. Combinations of radar and camera systems are promising [15], together with smaller radar devices for mid-sized UAVs [3] where ranges of hundreds of meters to few kilometers are reached depending on the size of the detected intruder. For small drones with very limited payload and power capabilities, cameras seem to be still the sensors of choice for this task [5].

Next to the best-case scenario with rather simple detection of dark pixels in front of blue sky, one criteria to detect intruders in camera images is their visually distinctive movement compared to the background which mainly represents a stationary scene. Hence, a majority of solutions focus on changes in the subsequent images after the sequence has been stabilized to perform background subtraction. Successful flight tests results have been presented in [13] where encounters between a ScanEagle UAV and a Cessna 172R are evaluated. Earlier versions of the algorithm are presented in [12]. The method consists of the pre-processing steps image stabilization and background subtraction, and then spatial (esp. morphological) and temporal filtering of image differences. Remaining features are then an indicator for regularly moving image regions as originated from other aircraft, and the method works also when the aircraft visually appears below horizon, see [10] for further details. The work in [16] is similar but focused on complex backgrounds, since image stabilization gives main benefit in such image regions. A different method based on multi-frame phase correlation is presented in [1] which seems to be very fast but comes with decreased robustness against complex backgrounds and highly dynamic background movements. The method presented in [14] differs between simple and complex background (means mostly above and below horizon) such that the detection method can be optimized depending on that background type.

III. Detection and Tracking Approach

The goal of this paper is to present a vision algorithm for detection and tracking of a small intruder against different kinds of backgrounds. The surveying camera is also moving since it is placed on a defending vehicle. The main challenges for the detection algorithm are the compensation of motion effects caused by the camera displacement and the inhomogeneous background. A secondary challenge for the algorithm is the fact, that the observed intruder aircraft can have a small size of just a few pixels in the image. Furthermore, in a typical real situation there are no information about the color or the shape of the intruding aircraft available. Fig. 1 shows a representative image captured by the defenders on-board camera. Therefore we concentrate on the detection of change caused by the movement of the object in the image.

The first step after image acquisition consists of image segmentation based on frame differencing followed by a spatial filter to detect candidate points for intruder in the image. In a second processing step, the candidate points are tracked over time, resulting in a temporal filtering process which keeps only candidate points which are successfully detected in several consecutive frames. Finally, the positions of the tracked candidate points are refined by a Kalman filter. This allows our algorithm to establish a robust tracking and to compensate for situations where the target is not detected in a frame, caused by several reasons. An overview of the proposed algorithms workflow can be seen in Fig. 2.

A. Image acquisition

Images are captured permanently during the flight from a forward looking camera on board of the defender with constant framerate of $25 fps$. In order to improve the quality of the following image processing steps, each image is undistorted as a first step directly after image acquisition. Undistortion is done based on distortion coefficients collected from an intrinsic camera calibration as a part of pre flight preparation.

B. Difference images

With no information about color or shape of the target UAV, we only try to look for motion indicated by scene changes in the stabilized image sequence where ego-motion has been removed. The next step in the image processing pipeline consists of calculating a difference image D_t from two consecutive frames I_{t-1} and I_t at time point t and $t-1$. Due to the movement of the camera, it is necessary to perform an image registration step on I_{t-1} to enable the calculation of a difference image. Common methods to achieve this task use feature detectors like Shi-Tomasi-Detector [7] or SURF-Detector [8] and estimate a homographic transformation between matched feature points. Still, these detectors can fail in finding good features or make mistakes in the alignment and matching of the features. Annother problem results in the fact that good feature points are often found in image regions with a high local contrast and are not distributed uniform in the image plane. Fig. 3 shows the distribution of detected feature points in the image plane summed up for 250 consecutive frames.

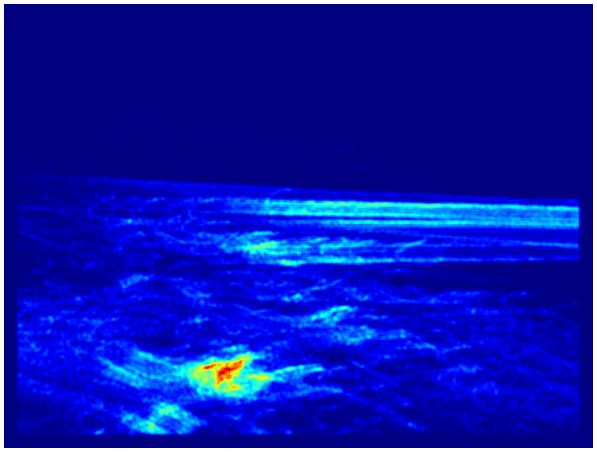


Fig. 3: Feature distribution for 250 frames.

With regard to the intended implementation on a companion computer with low computational power, avoiding the procedure of detection and mapping of feature points in the image speeds up the entire algorithm. Tests showed, that the presented method needs only 35% of the computational time, compared to feature detection based homography estimation.

The homographic transformation $H_{t-1\to t}$ for frame I_{t-1} is calculated based on the optical flow of a set of fixed grid points from the image. We select a sparse grid G of points from the image I_{t-1} and calculate the optical flow for each point $p \in G$ of the grid from I_{t-1} to I_t. An example of grid points (green) and their optical flow vectors (lime green arrows) can be seen in Fig. 4. In our experiments we found that a grid of size of 2067 grid points (53×39 points) for which the optical flow vectors are computed, works fine for a resolution of 1360 x 1024 pixel. Increasing the number of grid points will simultaneously increase the computational time for that step, while using to few points will decrease the accuracy of the transformation. The estimation of

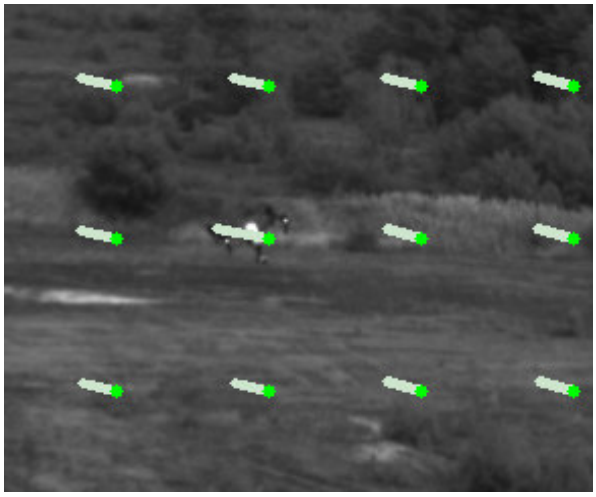


Fig. 4: Optical flow of selected grid points.

the optical flow is implemented with OpenCV's method *cv::calcOpticalFlowPyrLK* using the method of Lucas and Kanade [9] with a fixed window size of 21×21 pixel. Finally, I_{t-1}^{warped} is created by applying $H_{t-1\to t}$ on I_{t-1}.

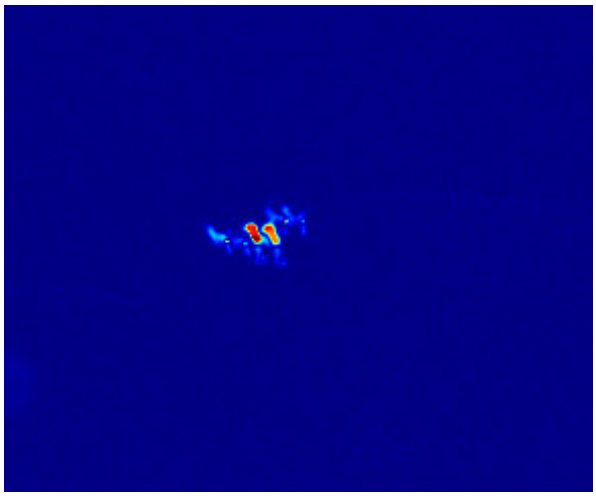


Fig. 5: Result of the frame differencing.

Next, the pixelwise absolute difference D_t between I_t and the warped image I_{t-1}^{warped} is calculated to achieve that areas with similar light intensity in the image will be set close to zero, while areas with a difference in the light intensity will be highlighted. Using the absolute difference makes the method invariant and robust against the illumination of the scene, so that different background structures or the unknown color of the intruder will have no effect. Fig. 5 shows the color coded results of this operation on two consecutive frames. Red and yellow areas have a high illumination change, while blue areas have lower change.

C. Spatial filtering

In the next steps, candidate points in the subtracted image that represent the intruder are extracted. To obtain a binary

segmentation, the subtracted image D_t is binarised using OpenCVs built-in function for adaptive threshold. Using adaptive threshold makes the algorithm robust on different local illumination conditions. From the binary image, a list of possible candidates for detection is extracted by identifying connected components in the image. The center point $c_{(x,y)}$ of each component is stored in a list $P_t^{candidates}$ and will be processed by the temporal filter in the next step.

D. Temporal particle filtering

Calculating candidate points only makes the algorithm not very robust to effects like motion blur or change of the size of the segmented objects (and also the position of their center point) in the image. Therefore we subsequently processed the list of possible candidate points with a temporal particle filter. This filter assigns a life time value τ to each candidate point, which is initialized with 1. For each time step t, the list $P_t^{candidates}$ of possible candidate points is compared with a list $P_t^{tracked}$ of already detected and tracked points. For each tracked point $p \in P_t^{candidates}$ the temporal filter checks if there is any possible candidate point $p \in P_t^{candidates}$ next to it. If so, c will be assigned to p and the life time value τ of p is increased by one. If no candidate point c was assigned to p, τ will be decreased by 1 until it is zero. Tracked points with life time value $\tau = 0$ will be deleted from the list of tracked points. Candidate points c which are not assigned to an already known point from $P_t^{tracked}$ at the end of this process are added to $P_t^{tracked}$ as new candidate points with $\tau = 1$.
Because we observed only one target UAV during our experiments, only the candidate point with the highest life time value τ was considered as a UAV.

E. Optimization with Kalman filter

Due to several reasons the estimated center position of each connected component $c_{(x,y)}$ and therefore also the candidate points can fluctuate from frame to frame. To compensate this effect, the position $c_{(x,y)}$ of each tracked point is corrected by a Kalman filter. The Kalman filter estimates the x and y position of the detected object in the image and additional the two-dimensional optical speed of the object in x and y direction, resulting in a 4-dimensional state space.
Keeping the positions updated by the Kalman filter makes the algorithm more robust against situations where the segmentation or tracking will be lost for a few frames. Furthermore, the correction of the UAV position by the Kalman filter results in a refinement of the 2D image trajectory and accuracy of the method.

IV. Experimental Results and System Evaluation

A. Experimental Setup

In the flight experiments, eight data sequences were recorded on two consecutive days with same weather conditions. The defender was represented by DLR's unmanned helicopter ARTIS (Fig. 6), a SwissDrones SDO-50 V2 with 85 kg MTOW and two 2.8 m intermeshing rotors. Most of the time during the experiments, the helicopter was operated remotely by a ground control station. Only takeoff and landing maneuver have been flown by a safety pilot. The helicopter was flown above a flat area free from obstacles, hovering in a position of about 30 m altitude facing in a fixed direction. The flight duration range from 7 to 10 minutes from take-off to landing for each flight. The used camera was an AVT GT-1380 producing grayscale images of 1360 $\times$ 1024 pixels at a frequency of 25 Hz. The optic is a 4.8 mm RICOH lens for the first four experiments and 10.0 mm Cinegon lens for the other four experiments.

Fig. 6: DLR's unmanned helicopter used as defender.

The intruder in our experiments was represented by a DJI Inspire Mark 1 (Fig. 7), flown in manual mode by a second pilot for the whole experiments. This UAV has a size of 44 $\times$ 45 $\times$ 38 cm, with a maximum top speed of 22 m/s at a weight of about 2.9 kg, for details see [6]. During the experiment, the intruder was performing different maneuvers in order to cover a wide variety of movements. The maneuvers and their effects on the image of the intruder on the camera plane are described later in section IV-B. Fig. 8 shows the scenario

Fig. 7: DLR's DJI Inspire 1, representing the intruder.

Fig. 8: Flight scenario for the experiments.

for all eight experiments. Blue area is the flight area of the observer, hovering in one position while the red area represents the fligh area of the intruder. The green area was used as a safety area, reserved for flight operation personal.

B. Image Sequences

From the recorded flight data we extracted eight different 10 sec long sequences to cover a wide combination of different imaging situations. Image sequence where selected by the following criteria: camera optic used during the flight experiments, movement of the intruder relative to the image plane and structure of the background area around the intruder. For the movement, we differentiate between scaling and translational movement. Scaling movement occurs when the distance between intruder and defender is increased or decreased, resulting in a change of the size of the intruder on the image plane. Translational movement describes a movement parallel to the camera image plane along the x or y axis.
Another differentiation in the sequences is made by the structure of the background of the area around the intruder. We discriminate between simple background, e.g. when the intruder is visible against the bright sky, or complex background, e.g. when the intruder is in front of structured background like trees. Table I gives a detailed overview on the sequences, including the background type (Bkgd), camera focal length F, distance between intruder and defender and the expected size of the intruder on the image plane.

TABLE I: Test sequences

Name	Movement	Bkgd	F (mm)	Size (px)	Dist. (m)
S1SS	scaling	simple	4.8	3 - 7	56 - 94
S1SC	scaling	complex	4.8	9 - 13	28 - 37
S1TS	translational	simple	4.8	21 - 22	16 - 21
S1TC	translational	complex	4.8	34 - 43	8 - 10
S2SS	scaling	simple	10.0	10 - 12	56 - 60
S2SC	scaling	complex	10.0	5 - 6	131 - 145
S2TS	translational	simple	10.0	18 - 26	43 - 53
S2TC	translational	complex	10.0	26 - 33	23 - 25

To estimate the real position of the two aircraft during the experiment, GPS data have been logged during the whole flight on both aircraft. Based on these logfiles, the distance between the aircraft for each single frame could be calculated. Together with the intrinsic parameter of the camera, resulting of the calibration of the camera, we were able to calculate the size of the intruder on the camera's image plane. To get the size of the intruder in the image, we calculated the projection of a simple box of same size as the intruder in the same distance from the camera onto the cameras image plane.

C. Precision Evaluation

In order to create a reliable set of ground truth data, the real position of the intruder was marked by hand for all frames in all sequences. Each single frame of a sequence was presented to a user who marked the x- and y-position of the

intruder in the image by clicking on its position. This position was stored and later compared to the output position of the detection algorithm to estimate the precision of the detection algorithm. Fig. 9 shows the the hand marked position of the intruder (green circle) and the position detected by the algorithm (blue cross). For each frame from a sequence, the Euclidean distance (in pixel) between the hand marked position and position detected by the algorithm was computed.

Fig. 9: Hand marked and estimated position of the intruder.

Due to the fact that even a trained user will not mark the same spot of the intruder in every image and the center of the detected intruder can vary from the hand marked ground truth, it is not easy to get a good rate for the detection. This holds especially for sequences where the intruder is close to the defender, so its area on the cameras image plane is bigger or sequences where the intruder is far away from the defender and is harder to mark by the user. To get a rate for the algorithms quality, we checked if the distance between the hand labeled position and the output of our algorithm was smaller than the diameter of the intruders projection on the image plane (see section IV-B for details on the projection). If so, the intruder was designated as detected and tracked. The percentage of frames with successful tracking for all experiments can bee found in table II, revealing a good quality from 85.0% up to 97.6%. The table also includes the time until the intruder was detected.
The cartesian deviation between the calculated position of the intruder and the ground truth position for each frame on all experiments can be seen in Fig. 10. The evaluation showed, that in most of our scenarios the intruder was detected successful after at most 25 frames. With a frame rate of $25 fps$, this results in a time of maximal one second until the algorithm has detected a target precisely. The timepoint of one second or 25 frames is marked with a dotted vertical line Fig. 10. Furthermore our experiment showed, that once the intruder is detected, the algorithm is able to track the intruder in the image for the whole remaining part of the sequence.

TABLE II: Test sequence evaluation

Name	Accuracy (%)	Detection time (s)
S1SS	85.6	0.2
S1SC	89.0	1.0
S1TS	97.6	0.2
S1TC	97.2	0.2
S2SS	85.0*	1.7
S2SC	91.6	0.2
S2TS	97.0*	0.2
S2TC	91.6	0.8

* values estimated

D. Performance Evaluation

The whole image processing pipeline was written in $C++11$ using the OpenCV library version $2.4.11$ for image processing. This configuration allows to port and test the software easily on different architectures. As mentioned before, one of the design criteria of the presented image analysis algorithm was the operation on companion computer onboard of the UAVs. To evaluate this ability, the algorithm was tested on three different hardware platforms, a $I7$ Workstation, a Nvidia Jetson TX1 and a Raspberry Pi Model 3. Table III gives an overview of the hardware which was used for this tests. A benchmark time for each filter per frame was measured using internal high resolution clock.

TABLE III: Hardware configuration

Hardware	CPU Type	Cores	Clock rate	Time(s)
Workstation	Intel I7 Skylake	8	3.4 Ghz	0.11
Jetson TX1	ARM Cortex A57	4	1.9 Ghz	0.44
Raspberry PI	ARM Cortex A53	4	1.2 Ghz	1.30

An example on the runtime for each component of the algorithm during a whole image sequence is shown in Fig. 11 for each hardware platform. The example is taken from sequence $S1SS$, but the algorithm shows the same runtime behavior on each platform for all the other sequences. One can observe, that the runtime for for each single component of the algorithm is almost linear at each timepoint, showing no dramatic increase at any point of the sequence. Still, the hardware platforms with ARM architecture showed some minor differences in the runtime of some components relative to each other, compared to the workstation. For the Nvidia Jetson TX1, one has to keep in mind that during all tests no GPU acceleration was used.

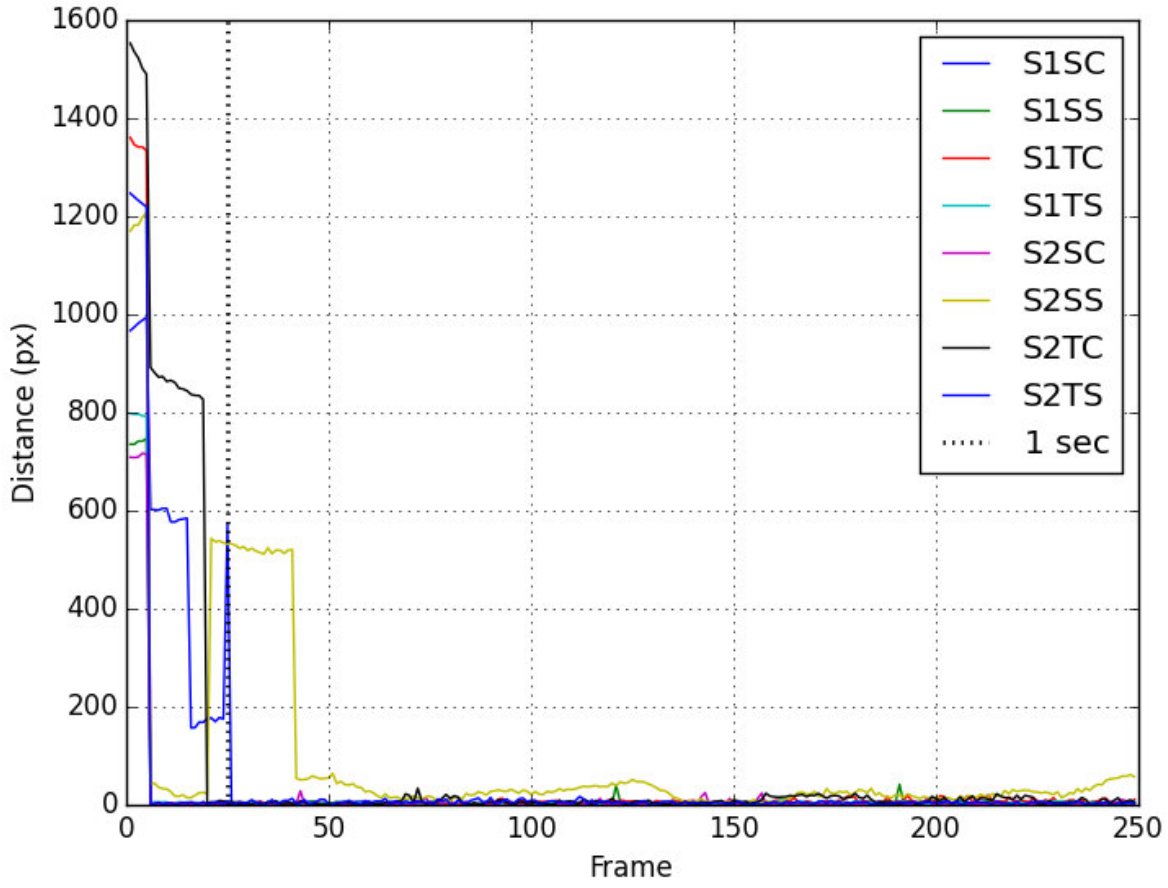


Fig. 10: Performance on all eight test sequences.

V. Conclusion and Outlook

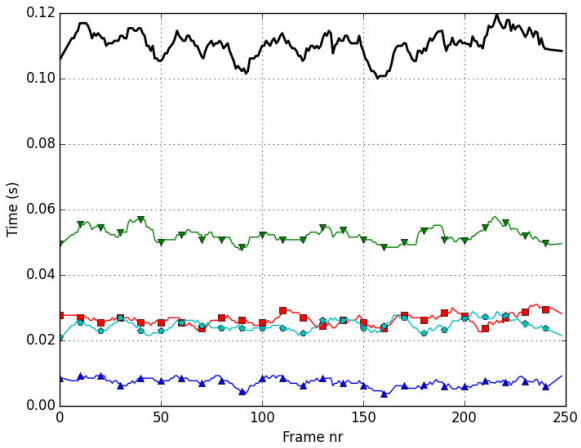
This paper describes a lightweight and fast algorithm for automatic detection and tracking of small moving objects from a moving camera, which is robust against different combinations of movement of the intruder and different type of backgrounds. The evaluation against a hand labeled ground truth showed, that the intruder can be detected in a reasonable time and will be tracked successful. A noteworthy effect of the algorithm is the ability to detect and track a single intruder without any previous selection or highlighting of the intruder, especially because there was no information or assumption about the shape or color of the intruder.

For the computational speed, there is still some space for optimization on different parts of the algorithm. Especially on companion computers with low computational power there is still some need for further optimization. Possible solution to that problem can result in decreasing the frame rate or the image size, resulting in the disadvantage of lowering the detection speed or resolution. From the selected hardware platforms, the Nvidia Jetson TX1 showed a big potential to be used as a companion computer, particular if the GPU unit from the board can be successful integrated in the image processing pipeline.

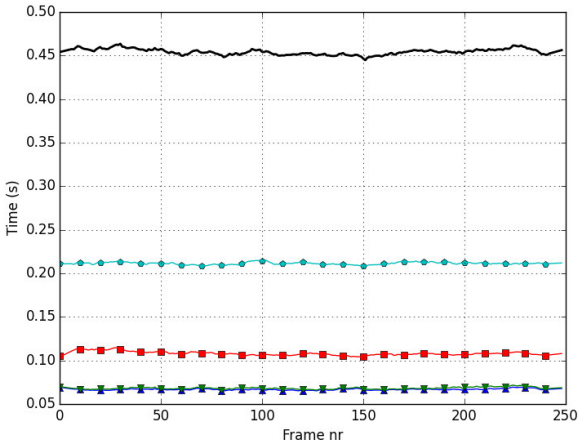
The presented algorithm was designed with regard to detect small moving object, which applies to UAVs as well as to birds. At the moment, there is no way to separate a detected object into one of these two classes. Further development should try to find a good solution do distinguish between birds and UAVs. One possible idea could be to analyze the trajectory of a tracked object in order to distinguish between UAVs and birds.

As mentioned in section IV-A, during the experiments only one intruder UAV was used. Therefore, the algorithm had to find and track only one target at all times. A future expansion could be the ability to track multiple intruder at the same time.

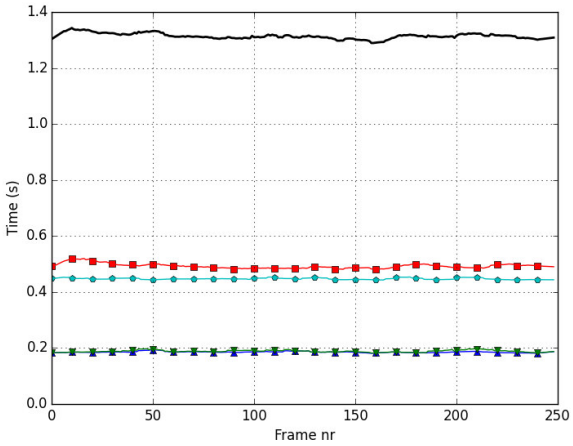
Nevertheless, the selected image sequences from the experiments contain a lot of different scenarios with different types of backgrounds, movements or sizes of the intruder. In combination with the hand-labeled ground truth positions, this yields in a good database for further development and investigations.



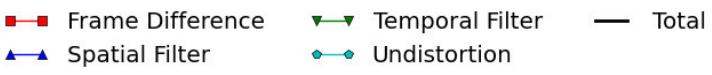
(a) Workstation



(b) Jetson TX1



(c) Raspberry PI 3 Model B



(d) Legend

Fig. 11: Runtime performance

REFERENCES

[1] F. Schubert, K. Mikolajczyk, "Robust registration and filtering for moving object detection in aerial videos," in *Pattern Recognition ICPR*, 22nd International Conference 2014, pages 2808-2813

[2] *OpenWorks Engineering*, https://openworksengineering.com/skywall, 2018

[3] *Echodyne Inc.*, www.echodyne.com [Online], 2018

[4] *TAR Ideal Concepts Ltd*, http://www.tarideal.com/Solutions/ANTI_DRONE_SOLUTIONS/Anti_Drone_Solutions/TA05066/DRONE_JAMMING_VEHICLE, [Online], January 2018

[5] B. C. Karhoff, J. Limb, S. W. Oravsky, A. D. Shephard, "Eyes in the domestic sky: an assessment of sense and avoid technology for the armys warrior unmanned aerial vehicle", in *Systems and Information Engineering Design Symposium*, 2006 IEEE, pages 36-42.

[6] DJI, "Inspire 1 User Manual, v2.0," Nov. 2015. [Online]. Available: dl.djicdn.com/downloads/inspire_1/en/Inspire_1_User_Manual_en_v2.0_1218.pdf

[7] J. Shi et al, "Good features to track," in *Computer Vision and Pattern Recognition*, Proceedings CVPR 94, IEEE Computer Society Conference on, pages 593-600. IEEE, 1994.

[8] D. ,G. Lowe, "Object recognition from local scale-invariant features," in *Computer Vision*, 1999. The proceedings of the seventh IEEE international conference on, volume 2, pages 1150-1157. Ieee, 1999.

[9] J. Y. Bouguet, "Pyramidal Implementation of the Lucas Kanade Feature Tracker", Microprocessor Research Labs, Intel Corporation, 2000

[10] T. L. Molloy, J. F. Fors, L. Mejias, "Detection of aircraft below the horizon for vision-based detect and avoid in unmanned aircraft systems," in *Journal of Field Robotics*, 2017

[11] G. Bradski, "The OpenCV library," in *Doctor Dobbs Journal*, Vol 25, 122–125. 2000

[12] J. Lai, L. Mejias, J. J. Ford, "Airborne vision-based collision-detection system", in *Journalof Field Robotics*, 28(2):137-157, 2011.

[13] D. Bratanov, L. Mejias, J. J. Ford, "A vision-based sense-and-avoid system tested on a scaneagle uav", in *In Unmanned Aircraft Systems (ICUAS)*, 2017 International Conference on, pages 1134-1142. IEEE, 2017.

[14] T. Zsedrovits, P. Peter, P. Bauer, B. J. M. Pencz, A. Hiba, I. Gozse, M. Kisantal, M. Nemeth, Z. Nagy, B. Vanek, et al., "onboard visual sense and avoid system for small aircraft", in *IEEE Aerospace and Electronic Systems Magazine*, 31(9):18-27, 2016

[15] G. Fasano, D. Accardo, A. E. Tirri, A. Moccia, E. De Lellis, "Sky region obstacle detection and tracking for vision-based uas sense and avoid.", in *Journal of Intelligent & Robotic Systems*, 84(1-4):121-144, 2016.

[16] S. Huh, S. Cho, Y. Jung, D. H. Shim, "Vision-based sense-and-avoid framework for unmanned aerial vehicles.", in *IEEE Transactions on Aerospace and Electronic Systems*, 51(4):34273439, 2015.